IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Boaz Ben-Zvi	Examiner: Pierre Michel Bataille
Serial No.:	10/677,398	Group Art Unit: 2186
Filed:	October 1, 2003	Docket No.: 200308873-1
Title:	Non-Blocking Distinct Grouping of Database Entries With Overflow	

APPEAL BRIEF UNDER 37 C.F.R. § 41.37

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief is filed in response to the Final Office Action mailed September 22, 2006 and Notice of Appeal filed March 20, 2007.

AUTHORIZATION TO DEBIT ACCOUNT

It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to Hewlett-Packard Development Company's deposit account no. 08-2025.

I. REAL PARTY IN INTEREST

The real party in interest is Hewlett-Packard Development Company, LP. a limited partnership established under the laws of the State of Texas and having a principal place of business at 20555 S.H. 249 Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

There are no known related appeals or interferences known to appellant, the appellant's legal representative, or assignee that will directly affect or be directly affected by or have a bearing on the Appeal Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1 – 12 stand finally rejected. The rejection of claims 1 – 12 is appealed.

IV. STATUS OF AMENDMENTS

No amendments were made after receipt of the Final Office Action. All amendments have been entered.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element or that these are the sole sources in the specification supporting the claim features.

Embodiments are generally directed to apparatus and methods for analyzing and returning data from a database. With exemplary embodiments, data retrieved from the database does not need to be grouped as aggregates before being transmitted to a user. Instead, data is processed using a hash grouping device that can return rows of data to the user substantially concurrently with the rows being received at the group-by node.

Claim 1 is directed to an apparatus that comprises a non-blocking grouping mechanism (#102) that groups entries of data (#104) and returns distinct entries of data substantially concurrently with processing following grouping of data (paragraphs [0021 – 0024 and 0028 – 0030]).

Claim 5 is directed to a method of providing concurrent grouping. The method comprises receiving input entries of data (paragraphs [0009] and [0014]); filtering out recurring entries of data from the input entries of data (paragraphs [0022 – 0023]); and returning distinct entries of data from the input entries of data to the user substantially concurrently with the receiving input entries of data (paragraphs [0023] and [0029 – 0030]).

Claim 9 is directed to a method of grouping entries of data. The method comprises prior to a potential overflow within a primary memory, grouping each input row of data and returning the data in a non-blocking fashion (paragraphs [0007 – 0010] and [0021 – 0022]); and in case of the overflow, ensuring that the user eventually receives the correct remaining rows (paragraphs [0025 – 0026] and [0031 – 0033]).

Claim 10 is directed to a method of grouping entries of data. The method comprises segmenting the groups into clusters that limit a potential overflow to one cluster at a time (paragraphs [0026 – 0028] and [0033]); prior to the potential overflow, all clusters perform work in a non-blocking fashion (paragraphs [0026 – 0030]); and in case of the overflow, transferring clusters one at a time from the primary memory to the secondary memory, while the remaining non-transferred clusters can still function in a non-blocking fashion (paragraphs [0025 – 0026] and [0031 – 0033]).

Claim 11 is directed to a method of grouping entries of data. The method comprises prior to a potential overflow within a primary memory, grouping each input row of data and returning the data in a non-blocking fashion (paragraphs [0026 – 0033]); and in case of the overflow in which at least some of the data is transferred from the primary memory to a secondary memory, this data on the secondary memory is later processed in a non-blocking fashion concurrently with processing the remaining data (paragraphs [0025 – 0026] and [0031 – 0033]).

Claim 12 is directed to an apparatus that comprises a non-blocking grouping mechanism (#102) that groups entries of data (#104), and returns distinct entries of data substantially concurrently with processing following entries of data to be grouped; (paragraphs [0021 – 0024] and [0028] – [0030]); an overflow mechanism by which data that includes the groups of entries of data that were grouped by the non-blocking grouping mechanism can be written from a primary memory (#512) to a secondary memory (#516) when the primary memory reaches an overflow condition (paragraphs [0025 – 0026] and [0031 – 0033]); and a return mechanism by which the data can be returned from the secondary memory back to the primary memory (paragraphs [0025] and [0029]), and whereupon the data is being returned to the user substantially concurrently with the rest of the data being processed by the non-blocking grouping mechanism (paragraphs [0010] and [0030]); a select mechanism by which a prescribed number of output groups are requested by the user, wherein operation of all of the non-blocking grouping mechanism, the overflow mechanism, and the return mechanism are halted when the requested prescribed number of output groups is reached (paragraphs [0009], [0034] and [0094]).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1 – 12 are rejected under 35 USC § 102(b) as being anticipated by USPN 5,511,190 (Sharma).

VII. ARGUMENT

The rejection of claims 1 – 12 is improper, and Applicant respectfully requests withdraw of these rejections.

The claims do not stand or fall together. Instead, Applicant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-heading as required by 37 C.F.R. § 41.37(c)(1)(vii).

Overview of Invention and Sharma

As a precursor to the arguments, Applicant presents a short overview of his invention and the primary reference Sharma.

Many databases are quite large and can contain millions of entries. When a user performs a query on such a large database, the number of computations to execute the query is also quite large. One goal is to execute the query and provide accurate results to the user as soon as possible so the user is not waiting a long period of time for the search results.

Applicant's invention and Sharma are both directed to executing database queries on large databases. The timing of when the search results are provided to the user is quite different in Applicant's invention and Sharma.

In Applicant's invention, search results to a database query are concurrently returned to the user while data from the database is continued to be processed. Data retrieved from the database does not need to be entirely grouped as aggregates before being transmitted to a user. Instead, data is processed using a hash grouping device that returns rows of data to the user substantially concurrently with the rows being received at the group-by node.

In Sharma, search results to a database query are not returned to the user until all the data is read and analyzed. Applicant's specification explains the shortcomings of Sharma:

> This prior-art version of the blocking hash groupby node is concentrated primarily on grouping aggregates of rows. With this

> prior-art version of the hash groupby node that is directed to aggregation, no single output row can be returned (to the user) before the last one of the input rows is read and processed. Unfortunately, such prior-art hash groupby nodes that rely on aggregate grouping typically demand a considerable amount of time to return any rows of data to the user. (See Applicant's specification at paragraph [0006]).

Claim Rejections: 35 USC § 102(b)

Claims 1 – 12 are rejected under 35 USC § 102(b) as being anticipated by USPN 5,511,190 (Sharma). These rejections are traversed.

Each independent claim recites at least one element not taught in Sharma. Some examples are provided below.

Claim 1

Claim 1 recites that data entries are returned "substantially concurrently with processing following grouping of data." In other words, claim 1 recites when data entries are returned. Data entries are returned substantially concurrently with processing. In direct contrast to claim 1, Sharma expressly teaches that data entries are not returned substantially concurrently with processing. In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed.

Applicant's interpretation of Sharma is supported in Sharma's specification. Figure 3 in Sharma provides a flow diagram of a hash grouping method for retrieving data and providing it to a user. Block 325 shows when in time results are provided to the user. Specifically, results are not provided to the user **until the last row of the input file is read**. Sharma discusses figure 3 and explains when results are reported to the user:

> [T]he grouping function tests whether the last row of the input file T1 212 has been read (325). If the end of the table T1 212 has not been reached (325 - N), the input procedure 232 of the grouping function GF 124a begins processing the next row of the input table (312). If the

> end of the table T1 212 has been reached (325 - Y), the contents of the group table are reported (327) to the user via the communications interface 114. (See Sharma col. 11, lines 32-40).

Thus, claim 1 clearly recites an element not taught in Sharma. In claim 1, data entries are returned "substantially concurrently with processing following grouping of data." By contrast in Sharma, data entries are not returned to the user until all data is read (i.e., until the last row of the input file is read).

Anticipation under section 102 can be found only if a single reference shows exactly what is claimed (see *Titanium Metals Corp. v. Banner*, 778 F.2d 775, 227 U.S.P.Q. 773 (Fed. Cir. 1985)). For at least these reasons, claim 1 and its dependent claims are allowable over Sharma.

Claim 5

Claim 5 recites "filtering out recurring entries of data from the input entries of data." Nowhere does Sharma teach that recurring entries of data are filtered out from the input entries of data. The Examiner cites Sharma at col. 2, line 54 to col. 3, line 6. Applicant respectfully disagrees.

Column 2, line 54 to column 3, line 6 in Sharma discloses a grouping method that reads rows of a database table, applies a hashing function that generates an index to a hash table, and then determines from the contents of the hash table whether a group table entry exists that corresponds to an identifier of a row. Nowhere does this section of Sharma even address filtering out recurring entries from input entries. In fact, Sharma does not even discuss filtering entries.

For a prior art reference to anticipate under section 102, every element of the claimed invention must be identically shown in a single reference (see *In re Bond*, 910 F.2d 831, 15 U.S.P.Q.2d 1566 (Fed. Cir. 1990)). For at least these reasons, claim 5 and its dependent claims are allowable over Sharma.

As another example, claim 5 recites "returning distinct entries of data from the input entries of data to the user substantially concurrently with the receiving input entries of data." Nowhere does Sharma teach that data entries from input data entries of data are returned substantially concurrently with receiving input entries. In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed (see Sharma col. 11, lines 32-40: data entries are not returned to the user until the last row of the input file is read).

Anticipation is established only when a single prior art reference discloses each and every element of a claimed invention united in the same way (see *RCA Corp. v. Applied Digital Data Systems, Inc.*, 730 F.2d 1440, 1444 (Fed. Cir. 1984)). For at least these reasons, claim 5 and its dependent claims are allowable over Sharma.

Claim 9

Claim 9 recites "prior to a potential overflow within a primary memory, grouping each input row of data and returning the data in a non-blocking fashion." In other words, claim 9 recites that before an overflow occurs within a primary memory, data is grouped and **returned in a non-blocking fashion**. Nowhere does Sharma teach that data entries are grouped and returned in a non-blocking fashion prior to an overflow within primary memory.

In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed (see Sharma col. 11, lines 32-40: data entries are not returned to the user until the last row of the input file is read).

Further, Applicant's specification explains the difference between blocking nodes (example, used in Sharma) and non-blocking nodes (recited in claim 9):

> With blocked nodes, data is not output from the node to the user until all of the data is read and analyzed. Nodes suited for aggregating rows of data are inherently blocked. With non-blocked nodes, individual rows of data that match a query can be output to the user prior to the node receiving all of its data (and the groups of data can be processed prior to reading all of the data in the database). (See Applicant's specification at paragraph [0010]).

There can be no difference between the claimed invention and the cited reference, as viewed by a person of ordinary skill in the art (see *Scripps Clinic & Research Foundation v. Genentech Inc.*, 927 F.2d 1565, 1576 (Fed. Cir. 1991)). For at least these reasons, claim 9 is allowable over Sharma.

Claim 10

Claim 10 recites "segmenting the groups into clusters that limit a potential overflow to one cluster at a time." In other words, the claim recites that groups of data are segmented into clusters to limit a potential overflow to **one cluster at a time**. Nowhere does Sharma disclose that groups of data are segmented into cluster so as to limit a potential overflow to "one cluster at a time." The Examiner cites three sections in Sharma (namely, 6: 32-44, 8: 24-42, and 11: 13-20). Applicant respectfully disagrees.

Column 6, lines 32-44 in Sharma discusses an overflow table in secondary memory that provides the input to a grouping function once the entire table is read. Nowhere does this section of Sharma discuss that groups of data are segmented into clusters to limit a potential overflow to **one cluster at a time**.

Column 8, lines 24-42 in Sharma discusses setting a flag when a grouping function accommodates data during an overflow. Nowhere does this section of Sharma discuss that groups of data are segmented into clusters to limit a potential overflow to **one cluster at a time.**

Column 11, lines 13-20 in Sharma discusses using overflow procedures if no memory is available for new group table entry. Nowhere does this section of Sharma discuss that groups of data are segmented into clusters to limit a potential overflow to **one cluster at a time.**

It is well settled that to anticipate a claim, the reference must teach every element of the claim, see M.P.E.P. § 2131. For at least these reasons, claim 10 is allowable over Sharma.

As another example, claim 10 recites "prior to the potential overflow, all clusters perform work in a non-blocking fashion." Nowhere does Sharma teach that clusters perform work in a non-blocking fashion. Sharma uses a blocking fashion.

In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed (see Sharma col. 11, lines 32-40: data entries are not returned to the user until the last row of the input file is read).

Further, Applicant's specification explains the difference between blocking nodes (example, used in Sharma) and non-blocking nodes (recited in claim 10):

> With blocked nodes, data is not output from the node to the user until all of the data is read and analyzed. Nodes suited for aggregating rows of data are inherently blocked. With non-blocked nodes, individual rows of data that match a query can be output to the user prior to the node receiving all of its data (and the groups of data can be processed prior to reading all of the data in the database). (See Applicant's specification at paragraph [0010]).

It is well settled that to anticipate a claim, the reference must teach every element of the claim, see M.P.E.P. § 2131. For at least these reasons, claim 10 is allowable over Sharma.

As yet another example, claim 10 "transferring clusters one at a time from the primary memory to the secondary memory, while the remaining non-transferred clusters can still function in a non-blocking fashion." Nowhere does Sharma teach this recitation. The Examiner cites three sections in Sharma (namely, 6: 32-44, 8: 24-42, and 11: 13-20). Applicant respectfully disagrees.

Column 6, lines 32-44 in Sharma discusses an overflow table in secondary memory that provides the input to a grouping function once the entire table is read. Nowhere does this section of Sharma discuss transferring clusters one at a time between memories while the remaining non-transferred clusters function in a non-blocking fashion. Sharma expressly teaches functioning in a blocking fashion.

Column 8, lines 24-42 in Sharma discusses setting a flag when a grouping function accommodates data during an overflow. Nowhere does this section of Sharma discuss transferring clusters one at a time between memories while the remaining non-transferred clusters function in a non-blocking fashion. Sharma expressly teaches functioning in a blocking fashion.

Column 11, lines 13-20 in Sharma discusses using overflow procedures if no memory is available for new group table entry. Nowhere does this section of Sharma discuss transferring clusters one at a time between memories while the remaining non-transferred clusters function in a non-blocking fashion. Sharma expressly teaches functioning in a blocking fashion.

It is well settled that to anticipate a claim, the reference must teach every element of the claim. see M.P.E.P. § 2131. For at least these reasons, claim 10 is allowable over Sharma.

Claim 11

Claim 11 recites "prior to a potential overflow within a primary memory, grouping each input row of data and returning the data in a non-blocking fashion." In other words, claim 11 recites that before an overflow occurs, data is grouped and **returned in a non-blocking fashion**. Nowhere does Sharma teach that data entries are grouped and returned in a non-blocking fashion prior to an overflow.

In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed (see Sharma col. 11, lines 32-40: data entries are not returned to the user until the last row of the input file is read).

Further, Applicant's specification explains the difference between blocking nodes (example, used in Sharma) and non-blocking nodes (recited in claim 11):

> With blocked nodes, data is not output from the node to the user until all of the data is read and analyzed. Nodes suited for aggregating rows of data are inherently blocked. With non-blocked nodes, individual rows of data that match a query can be output to the user prior to the node receiving all of its data (and the groups of data can be processed prior to reading all of the data in the database). (See Applicant's specification at paragraph [0010]).

There can be no difference between the claimed invention and the cited reference, as viewed by a person of ordinary skill in the art (see *Scripps Clinic & Research Foundation v. Genentech Inc.*, 927 F.2d 1565, 1576 (Fed. Cir. 1991)). For at least these reasons, claim 11 is allowable over Sharma.

As another example, claim 11 recites that data is processed in a "non-blocking fashion concurrently with processing remaining data." Nowhere does Sharma teach this recitation. The Examiner cites three sections in Sharma (namely, 6: 32-44, 8: 24-42, and 11: 13-20). Applicant respectfully disagrees.

Column 6, lines 32-44 in Sharma discusses an overflow table in secondary memory that provides the input to a grouping function once the entire table is read. Nowhere does this section of Sharma discuss data is processed in a "non-blocking fashion concurrently with processing remaining data."

Column 8, lines 24-42 in Sharma discusses setting a flag when a grouping function accommodates data during an overflow. Nowhere does this section of Sharma discuss data is processed in a "non-blocking fashion concurrently with processing remaining data."

Column 11, lines 13-20 in Sharma discusses using overflow procedures if no memory is available for new group table entry. Nowhere does this section of Sharma discuss data is processed in a "non-blocking fashion concurrently with processing remaining data."

There can be no difference between the claimed invention and the cited reference, as viewed by a person of ordinary skill in the art (see *Scripps Clinic & Research Foundation v. Genentech Inc.*, 927 F.2d 1565, 1576 (Fed. Cir. 1991)). For at least these reasons, claim 11 is allowable over Sharma.

Claim 12

Claim 12 recites a grouping mechanism that "returns distinct entries of data substantially concurrently with processing following entries of data to be grouped." Nowhere does Sharma teach that data entries are returned substantially concurrently with processing. In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed (see Sharma col. 11, lines 32-40: data entries are not returned to the user until the last row of the input file is read).

In order for a prior art reference to be anticipatory under 35 U.S.C. § 102 with respect to a claim, "[t]he elements must be arranged as required by the claim," see M.P.E.P. § 2131, citing *In re Bond*, 15 U.S.P.Q.2d 1566 (Fed. Cir. 1990). For at least these reasons, claim 12 is allowable over Sharma.

As another example, claim 12 recites that a return mechanism "whereupon the data is being returned to the user substantially concurrently with the rest of the data being processed by the non-blocking grouping mechanism." Nowhere does Sharma teach a return mechanism that returns data substantially concurrently with the rest of the data being processed. In Sharma, data entries are not returned to the user until all the data is read and analyzed. No single output row is returned (to the user) before the last one of the input rows is read and processed (see Sharma col. 11, lines 32-40: data entries are not returned to the user until the last row of the input file is read).

In order for a prior art reference to be anticipatory under 35 U.S.C. § 102 with respect to a claim, "[t]he elements must be arranged as required by the claim," see M.P.E.P. § 2131, citing *In re Bond*, 15 U.S.P.Q.2d 1566 (Fed. Cir. 1990). For at least these reasons, claim 12 is allowable over Sharma.

CONCLUSION

In view of the above, Applicant respectfully requests the Board of Appeals to reverse the Examiner's rejection of all pending claims.

Any inquiry regarding this Amendment and Response should be directed to Philip S. Lyren at Telephone No. 832-236-5529. In addition, all correspondence should continue to be directed to the following address:

Hewlett-Packard Company
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Respectfully submitted,

/Philip S. Lyren #40,709/

Philip S. Lyren
Reg. No. 40,709
Ph: 832-236-5529

VIII. Claims Appendix

1. An apparatus, comprising: a non-blocking grouping mechanism that groups entries of data, and returns distinct entries of data substantially concurrently with processing following grouping of data.

2. The apparatus of claim 1, further comprising an overflow mechanism by which data that includes the groups of entries of data that were grouped by the non-blocking grouping mechanism can be written from a primary memory to a secondary memory when the primary memory reaches an overflow condition.

3. The apparatus of claim 1, further comprising: an overflow mechanism by which data that includes the groups of entries of data that were grouped by the non-blocking grouping mechanism can be written from a primary memory to a secondary memory when the primary memory reaches an overflow condition; and a return mechanism by which the data can be returned from the secondary memory back to the primary memory, and whereupon the data is being returned to the user substantially concurrently with the rest of the data being processed by the non-blocking grouping mechanism.

4. The apparatus of claim 1, wherein the primary memory includes a primary Random Access Memory (RAM).

5. A method of providing concurrent grouping, comprising: receiving input entries of data: filtering out recurring entries of data from the input entries of data; and returning distinct entries of data from the input entries of data to the user substantially concurrently with the receiving input entries of data.

6. The method of claim 5, wherein the method accommodates memory overflow by selected portions of the entries of data in a primary memory being flushed to a secondary memory to alleviate memory pressure.

7. The method of claim 5, wherein the method accommodates a memory overflow, wherein clusters of entries of data are written from a primary memory to a secondary memory when the primary memory runs out of memory, and wherein the primary memory overflows into the secondary memory by flushing one of its clusters of entries of data into the secondary memory and releasing certain ones of its in-memory buffers.

8. The method of claim 5, further comprising returning entries of data in a non-blocking fashion concurrently with other entries of data being processed.

9. A method of grouping entries of data, comprising: prior to a potential overflow within a primary memory, grouping each input row of data and returning the data in a non-blocking fashion; and in case of the overflow, ensuring that the user eventually receives the correct remaining rows.

10. A method of grouping entries of data, comprising: segmenting the groups into clusters that limit a potential overflow to one cluster at a time; prior to the potential overflow, all clusters perform work in a non-blocking fashion; and in case of the overflow, transferring clusters one at a time from the primary memory to the secondary memory, while the remaining non-transferred clusters can still function in a non-blocking fashion.

11. A method of grouping entries of data, comprising: prior to a potential overflow within a primary memory, grouping each input row of data and returning the data in a non-blocking fashion; and in case of the overflow in which at least some of the data is transferred from the primary memory to a secondary memory, this data on the secondary memory is later processed in a non-blocking fashion concurrently with processing the remaining data.

12. An apparatus, comprising: a non-blocking grouping mechanism that groups entries of data, and returns distinct entries of data substantially concurrently with processing following entries of data to be grouped; an overflow mechanism by which data that includes the groups of entries of data that were grouped by the non-blocking grouping mechanism can be written from a primary memory to a secondary memory when the primary memory reaches an overflow condition; and a return mechanism by which the data can be returned from the secondary memory back to the primary memory, and whereupon the data is being returned to the user substantially concurrently with the rest of the data being processed by the non-blocking grouping mechanism; a select mechanism by which a prescribed number of output groups are requested by the user, wherein operation of all of the non-blocking grouping mechanism, the overflow mechanism, and the return mechanism are halted when the requested prescribed number of output groups is reached.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.